Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1-97. (Cancelled)

98. (Previously Presented) A method comprising:

receiving a first opcode executing in a first thread of execution;

translating a linear address associated with said first opcode into a physical address;

executing a bus transaction by a monitoring bus agent to ensure no other bus agent has sufficient ownership of data associated with said physical address to allow another bus agent to modify the data without informing the monitoring bus agent;

monitoring for an access to said physical address;

signaling a hit if another bus agent reads said physical address;

receiving a second opcode in the first thread of execution;

suspending said first thread of execution and enabling recognition of a monitor event in response to the second opcode;

resuming said first thread if the access occurs;

resuming execution of the first thread in response to any one of a first set of events.

99. (Previously Presented) The method of claim 98 further comprising:

detecting a second set of events; and

ignoring said second set of events differing from said first set of events.

100. (Previously Presented) The method of claim 98 wherein said access is a write access.

101. (Previously Presented) The method of claim 98 wherein said linear address corresponds to an address of a predetermined type of memory as a precondition to said first thread being suspended.

102. (Previously Presented) The method of claim 101 wherein said predetermined type of memory is write back memory.

103. (Previously Presented) The method of claim 98 wherein said bus transaction is a read bus transaction.

104. (Previously Presented) The method of claim 98 further comprising:

relinquishing a plurality of partitioned resources in response to the first thread being suspended;

partitioning a plurality of resources in response to the access.

105. (Previously Presented) The method of claim 104 wherein said plurality of partitioned resources comprise:

an instruction queue;

a re-order buffer;

a pool of registers;

a plurality of store buffers.

106. (Previously Presented) The method of claim 98 wherein suspending the first thread of execution in response to the second opcode comprises:

testing whether the monitor event is pending;

testing whether a monitor is active;

if the monitor is active and no monitor event is pending, then entering a first thread suspended state.

107. (Previously Presented) The method of claim 106 wherein entering the first thread suspended state comprises:

relinquishing a plurality of registers in a register pool;

relinquishing a plurality of instruction queue entries in an instruction queue;

relinquishing a plurality of store buffer entries in a store buffer;

relinquishing a plurality of re-order buffer entries in a re-order buffer.

108. (Previously Presented) A system comprising:

a memory to store a loop from a first thread, said loop including a first instruction, a second instruction and a test to determine whether data at a monitor address has changed and to restart said loop if said data at said monitor address remains unchanged, the first instruction having an associated address operand specified by an operand, the operand being an implicit operand in a predetermined register indicating said monitor address;

a first processor coupled to said memory, said first processor to enable a monitor to monitor memory transactions to detect a memory access to said monitor address in response to the first instruction and to cause resumption of said first thread in response to the memory access to the monitor address.

109. (Previously Presented) The system of claim 108 wherein said memory is to store a second instruction from said first thread, and wherein said first processor is to suspend said first thread in response to the second instruction.

110. (Previously Presented) The system of claim 109 wherein said monitor is to set a monitor event pending indicator in response the memory access occurring, said monitor event pending indicator to cause said first processor to resume a thread once unmasked by said second instruction.

111. (Previously Presented) The system of claim 108 wherein said first processor includes a first cache, the system further comprising:

a second processor comprising a second cache, wherein said first processor drives a bus transaction to the second processor to force said second processor to broadcast to the first processor any transactions that allow alteration of data stored at the monitor address in the second cache.

112. (Previously Presented) The system of claim 111 wherein said first processor is to assert a signal preventing said second processor from caching data at the monitor address in a state which would allow the second processor to modify data stored at the monitor address in the second cache without broadcasting that a modification is occurring.

113. (Previously Presented) The system of claim 112 wherein said signal indicates a cache hit and prevents the second cache from storing data at the monitor address in an exclusive state.

114. (Previously Presented) The system of claim 110 wherein said first processor is further to resume the first thread if an alternative event occurs.

115. (Previously Presented) The system of claim 114 wherein said alternative event is an interrupt.

116-121. (Cancelled).